~~Paper to be~~ Presented at the Sci~~entific~~ Data Systems (SDS) Users' Group Meeting, ~~November 15-16, 1963,~~ Las Vegas, Nev~~ada~~ 15-16 Nov. 1963

A DIGITAL LUNAR MIDCOURSE GUIDANCE SIMULATION

N65-88996 ~~X64 10749~~

By David Kipping* [1963]

31p ref

Code 2A

(NASA TM X-51287)

11749

ABSTRACT

A manned lunar midcourse guidance simulation is described. Actual hardware is used to simulate the onboard operations which are performed in a manned lunar mission. An optical simulation, accurate to a few seconds of arc, will provide optical input data from human observers. The space-borne digital computer is simulated by an SDS 920 computer system. The computer system includes a Spacecraft Computer Control and Display Panel from which the computer can be controlled in real time. The midcourse guidance scheme uses linear prediction and a statistical filter on the optical observations for the optimal estimation of ~~position and velocity of the spacecraft. The basic midcourse program is a long~~ program (10,000 words) which is written in FORTRAN for programming ease, but is as tightly optimized as FORTRAN will permit. The techniques for doing so are described. The rest of the computer's 12,000 words are devoted to SYMBOL programs which interchange information with the FORTRAN program, mainly through the use of interrupts. The programming aspects of the simulation, including the problem of controlling a FORTRAN program from a manual entry keyboard and control panel are discussed.

Author

INTRODUCTION

In order to do the navigation required for a manned lunar mission, a digital computer of large capacity and high speed is required to be onboard the spacecraft.

---

*Research Scientist, NASA Ames Research Center, Moffett Field, Calif~~ornia~~

To increase reliability and to give more flexibility to the navigation system, it is desirable for a human navigator or pilot to be able to control the computations done by the computer, and monitor the results. This paper outlines the hardware and programming techniques which allow this to be done in a lunar midcourse guidance simulation using an SDS 920 computer. The simulation uses both FORTRAN and machine language programs in a way which is unique for the SDS 900 series computers in that the FORTRAN program is controlled by programs which are initiated by interrupts. This technique gives the advantages of real time machine language programming using interrupts, and yet gives the ease of programming in FORTRAN.

DESCRIPTION OF SIMULATION EQUIPMENT

The Guidance and Control Systems Branch at Ames Research Center is setting up a laboratory for the investigation, from a hardware point of view, of guidance and control systems which have as a functional element an airborne or spaceborne digital computer (Figure 1). Navigation instruments, such as theodolites, inertial platforms, radars, and conventional aircraft navigational equipment can be investigated. For optical measurements, a very accurate simulation provides collimated celestial images moving relative to each other at controlled rates. The onboard computer is simulated by an SDS 920 computer system which can communicate with a pilot or operator as well as with external electronic equipment. Analog/digital conversion equipment is provided to communicate with an analog computer if the latter is needed to provide short period dynamics.

The SDS 920 computer system is shown in Figure 2. The computer has a complete complement of standard peripheral input/output devices, plus a wide variety of equipment to facilitate exchange of information from other electronic sources.

Summing registers and pulse rate generators allow handling of pulse rate information on either input or output, an address selector decodes 128 commands to select external devices, and the priority interrupt system is used extensively in controlling and accepting information from these devices. The real time clock system gives time of day accurate to one part in 10^7 and provides the computer with a time reference with a resolution of 0.01 seconds.

For the midcourse guidance simulation, the Spacecraft Computer Control and Display Panel (Figure 3) is of primary importance as it serves for data input and control of the guidance computations. In the upper left hand corner of the panel are status and warning lights, and in the upper right hand corner is the display panel for the real time clock, while the rest of the panel is associated with data input and display. The data format is illustrated by the thumbwheel input panel at the lower left, where the first four thumbwheels from the left (the Sense thumbwheels) provide an operation code to the computer, and the numeric thumbwheels represent a floating point number with an eight decimal digit mantissa and a one digit exponent. The Matrix Identification provides a two digit identification code for any purpose desired. Data is transmitted to the computer as binary-coded-decimal information for the numerical thumbwheels, and as a binary code for the alphabetic characters of the Sense thumbwheels.

The panel just above the thumbwheels is a series of push buttons which generate interrupts to start programs and above this is the electroluminescent alphanumeric display, arranged in the same format as the thumbwheels. It is used to display information and to confirm that data is properly entered into the computer.

This entire Control Panel allows a pilot to enter data, control the computations done with easily remembered mnemonics, and to monitor the results. It is designed and constructed so that it may be operated up to 200 feet from the computer proper. It is clear that thumbwheels and a keyboard are not both required for data entry, but it is desired to evaluate each approach from an operational viewpoint as well as from a hardware mechanization.

OPTICAL SIMULATION

An accurate optical simulation has been constructed to provde images for evaluation of optical instruments and to provide input data for the mission simulation. For both sextant or theodolite measurements, it is necessary to sight on a planet (earth or moon) moving relative to the fixed stars at appreciable rates. To simulate this motion a precision rate table is used (Figure 4). The image of the planet is formed and collimated with optics mounted on a rotary table driven by a synchronous motor. To change the angle the table is rotated and translated, thus maintaining an image at the observing station that is apparently moving across the sky. By using similar optics, one or more fixed stars can also be simulated.

The planet image can be moved at rates from 0.1 to 5.0 seconds of arc per second of time, typical rates for the midcourse phase of a lunar mission. Using a precision shaft angle encoder, the position of the rotary table can be measured to a few seconds of arc, and the collimation error is held to less than five seconds of arc over a six inch diameter viewing station. Thus, the simulation can produce adequate images for optical equipment, accurate to a few seconds of arc.

DESCRIPTION OF MIDCOURSE GUIDANCE SIMULATION PROGRAMS

An overall block diagram of the midcourse guidance simulation is shown in Figure 5. Using the optical simulation, the observer can make either a theodolite measurement of the position of a planet or of a landmark on the planet, or can make a measurement of the angle between a planet and a fixed star with a sextant. The exact time of the observation is introduced into the computer by pushing the Time Interrogate button which, using an interrupt, records the time in the computer. After the observation is made, the instrument dials are read and their value is an input to the computer via the Control Panel.

The onboard computer simulation consists of three main programs. The largest of the three is the Midcourse Guidance Program which computes the guidance equations and velocity correction parameters. The Computer System Program does the formating and logical operations required to enable the Control Panel, clock, and other computer system devices to exchange information with the computer. When the Execute button is pushed, control is transferred to the Midcourse Control and Display Program, the function of which is to convert the data to the proper form, store or retreive it from the Midcourse Guidance Program, and control operations of the Midcourse Guidance Program. As the velocity corrections are computed, two small programs simulate the operation of actually making a velocity correction.

As much as possible, these programs are written for maximum efficiency and do only the computations which would actually be done aboard a spacecraft. They are all written in SYMBOL (the SDS symbolic program assembler) with the exception of the Midcourse Guidance Program which, although is written in FORTRAN, is as tightly optimized as that language will permit.

In addition to the programming mentioned above, there are a number of computations associated with operating and evaluating the simulation, and would not actually be carried aboard a spacecraft. In order to evaluate the performance of the guidance system, the actual trajectory of the spacecraft is computed, a quantity which in practice would never be known. For control of the optical simulation, positioning and rate information for the optical table is computed and displayed, and the actual planet position at the instant of observation is measured with the shaft angle encoder on the precision optical table, and is fed back into the computer. The midcourse rocket engine and the sensing accelerometers are simulated by a small program. For evaluation of the simulation, considerable program space is devoted to typing out all the significant parameters as each computation is made.

The position of the Computer System Program which operates the Real Time Clock and performs event timing, is shown in Figure 6. Associated with the Computer System Program are a number of Registers. These are merely cells in memory to which a special name has been given for convenience. The Data Register is used for intermediate storage of Control Panel Sense information and numerical data in binary-coded-decimal form, the Time Register retains the running value of real time, and the Event Register is used to hold the time at which a specific event occurred for later use by the computer.

The Real Time Clock keeps time accurate to one part in 10^7 (which gives a maximum error of 0.1 second in 14 days) and can be started and stopped by push buttons which generate interrupts to perform these functions. Setting the Clock is accomplished with the Set push buttons which generate an interrupt to transfer the previously entered time in the Data Register into the Clock. The Time Register is initialized with the Set pushbutton, and is updated every 0.01 second by an interrupt generated by the Clock. This same 0.01 second interrupt may be used to initiate real time programs at any multiple of 0.01 seconds. Since it is possible that occasionally a few updating pulses may be lost, every minute the Clock generates another interrupt, which initiates a program to interrogate the Clock and transfer the complete time-of-day into the Time Register. If the Time Interrogate button is pushed, an interrupt is generated which transfers the instantaneous time from the Time Register in to the Event Register.

Also mechanized is a Countdown Register. This Register is first initialized to some value of time in seconds, counts down in increments of one second to zero and then counts up at the one second rate indefinitely. When the Start Countdown button is pushed, an interrupt is generated which initializes the Register by forming a time-to-go in seconds from a previously entered future time in the Data Register and the present time from the Time Register. Using a one second interrupt from the Clock, this time is counted down to zero and is then counted up. At zero, a signal is given to indicate the coincidence of future and present time. The contents of the Register in seconds are displayed on the alphanumeric display with a minus sign on the count down and a plus sign on the count up. This counting function may be terminated at any time by the Stop Countdown push button which generates an interrupt.

In Figure 7 is shown the data input and display portion of the System Program. Information may be loaded into the Data Register either by the parallel input thumbwheel switches, or by the serial input keyboard. The coding is such that no matter which input method is used, the final product in the Data Register is the same. For a thumbwheel input, the data is taken into the computer directly from the thumbwheels and is displayed on the alphanumeric display. For a keyboard input the keyboard is enabled by the Enter Keyboard button which generates an interrupt. The data keys are then pushed and the data is constructed character by character in the Data Register using the keyboard interpretive program which formats it correctly and displays the data after each character is inputted. The data keys generate one interrupt but each key has its own unique code, so the program must determine which key was pressed and,depending on the order, which character it represents. If an error is made or when keyboard use is no longer desired, the Release Keyboard button clears the Data Register and disconnects the keyboard. The Execute push button generates a separate high priority interrupt which transfers control immediately to the Midcourse Control and Display program.

The block diagram of the Midcourse Control and Display program is shown in Figure 8. The first four characters in the Data Register are the Sense characters. The leftmost one defines the mode of operation: data input, data output and display, and compute. The next three are a mnemonic operation code. After transferring into this program, the mode and operation code are translated and a transfer to the appropriate section of the program is made. For data input a number is taken from the Data Register, converted from BCD to binary, and stored in the appropriate place in COMMON in the Midcourse Guidance Program. For data output, the above procedure is reversed and the data is displayed from the Data Register. For a Compute operation, control flags are set in the Midcourse Guidance Program and control is transferred to that program.

If it is desired to make a velocity correction the computer can issue controlling signals to the rocket engine directly. One simple scheme for this is shown in Figure 9. It is assumed that the axis of the rocket engine is pointed in the right direction, and that this attitude is stabilized. From the computed velocity correction (in meters/sec.) a thrusting duration in seconds is computed. The future time of firing is entered into the Data Register and the countdown is started. As zero is detected, the rocket engine is commanded to fire. The up count is continually tested against the computed thrust duration and when that time is reached the Countdown Register is disabled, and a stop signal is sent to the rocket engine. The body-mounted accelerometer generates a pulse rate proportional to acceleration which is summed in the summing register to get total measured velocity correction. This information is then transferred into the Midcourse Guidance Program, and also could be used to control the stopping of the Countdown Register in case direct control of the firing duration is desired.

A simulation of the rocket engine and the accelerometer can be done largely with the pulse rate generator. The start signal enables the pulse rate generator, and a rate similar to the output of an accelerometer is produced. The stop engine signal is first delayed to represent the thrust taper-off of a rocket engine when it is turned off and then is used to stop the pulse rate generator. To compute the actual velocity correction a random number representing the error of the accelerometer is added to the measured velocity correction.

MIDCOURSE GUIDANCE PROGRAM

The heart of the simulation is the Midcourse Guidance Program. The guidance scheme used is that developed by the Theoretical Guidance & Control Branch at Ames Research Center. It has been presented elsewhere (References 1, 2 and 3) so only a brief summary of the theory is given here (Figure 10). Four body equations of motion for a reference and estimated trajectory are computed and integrated starting from injection conditions. A set of perturbation equations are solved for the state, which is a six-vector of differences between the estimated and reference positions and velocities. Observations made with sensors (optical instruments or accelerometers) are processed to obtain the optimum estimation of the dynamical state. Linear prediction is used to compute deviation from the desired end point and a suitable velocity correction is computed to reduce the position error at the end point to zero. The velocity correction is made but in practice the end point error is not reduced to zero due to errors in performing the correction and uncertainties in the estimated position and velocity. A schedule of observations and velocity corrections is chosen to reduce this error to an acceptable level.

The data processing scheme for optimal state estimation is shown in Figure 11. From the estimated state an estimate of the space angles observed is made. This is compared with the actual observations and the difference is multiplied by a weighting matrix to produce an improved estimate of the state. A matrix of the statistics of the estimation errors is computed and from it the weighting matrix is found by a process which takes into account the known error statistics of the sensors. This weighting matrix can be regarded as a dynamical time-varying filter which weights the observations in an optimal fashion.

The block diagram of the Midcourse Guidance Program is shown in Figure 12. The program actually has eleven different entry points but it has only three basic options: theodolite observation, sextant observation, or velocity correction. In all cases, the equations of motion are integrated to the time of the observation (or time of desired velocity correction) and the guidance matrices which are a function of time, are updated to the same time. Then the program branches either to process the input data or to perform the velocity correction.

PROGRAMMING TECHNIQUES

Programming for the Midcourse Guidance Program required by far the greatest amount of time in developing the midcourse guidance simulation. After a study of the effort involved, it was decided to write the program in FORTRAN because of the ease of writing in that language as opposed to writing in SYMBOL. In writing the program, the most important consideration was to achieve program efficiency. The SDS 920 configuration has 12,000 words of memory, about 8000 of which are allocated to the FORTRAN program. A major goal in programming was to minimize the length of the guidance program without losing any of the essentials of the mathematics. Several techniques were used to achieve this goal:

1. Optimization of Subroutines--Each subroutine was written to do only the minimum operations required. Computation techniques were carefully examined so that the calculations were performed in a minimum number of instructions. Whenever possible, subroutines were written with no call list at all in order to minimize the instructions required in setting up the transfer vector.

2. Data Storage--All data is stored in COMMON and a block is allocated as temporary storage for use by all programs. Only essential data is stored in COMMON while all other data that is not required between observations is discarded.

3. Minimization of Output--In the spacecraft computer simulation, all actual output is to displays and other hardware. However, for evaluation of performance a certain amount of typewriter output is desired. The output information was chosen to be a very minimum in order to shorten output routines.

4. Sun Moon Polynomial--In the computation of the equations of motion the positions of the sun and moon are required. One technique is to search a stored

table for points near the time desired and then do an interpolation. However, to eliminate the rather large stored tables and separate interpolation routines, a different technique is used. A tenth-order polynomial of position good for 14 days is evaluated with a simple polynomial evaluation routine for either sun or moon position or velocity. The coefficients for this polynomial are precalculated with a mathematical technique which minimizes the maximum error in a fashion similar to a Chebeshev polynomial, giving a fit good to within a few tenths of a kilometer.

The preliminary work was done on an IBM 7090 which resulted in an operating program but some of the optimization features mentioned above were not included. In Figure 13 is shown a comparison of storage requirements for the Midcourse Guidance program as implemented on the IBM 7090 and the SDS 920. For both computers, the column marked Simulation is the storage required for each computer to actually run the program, while the columns marked Onboard represent only those operations which would be done aboard a spacecraft. The figures for the 7090 are exact while the figures for the SDS 920 are only estimates derived from the 7090 figures, and may be a little high. From the figure it is seen that a considerable saving in storage is accomplished in the SDS 920 by use of all the techniques for program efficiency mentioned above. The last two items in Figure 13 are concerned with the FORTRAN run-time systems for the computers. The greater sophistication and input/output of the IBM 7090 is reflected in the approximately 5000 more words that it requires in its FORTRAN run-time system.

From the totals in the figure it is seen that the SDS 920 program is estimated to take about 10,000 words to implement. By elimination of some desirable features it would be possible to cut the program down to approximately the 8000 words allocated originally to the program. If all the operations that are associated only with the simulation are eliminated, the total onboard computations would take about 8100 words. If the program had not been written in FORTRAN and had been as tightly optimized in SYMBOL as possible, it is estimated that these computations would have taken about 6700 words.

SYSTEM INTEGRATION

In integrating the simulation together, one of the major problems is locating the various programs relative to each other and establishing communication between programs. The FORTRAN programs as they stand are not compatible with interrupt generated programs since a linkage table for the run time system occupys the interrupt locations. A program has been written which shifts this FORTRAN linkage table to a new location and changes appropriately all the instructions which refer to these locations. Then the interrupt locations can be filled with the proper branching instructions.

After the FORTRAN system and programs are loaded, the Computer System Program, the Midcourse Control and Display Program, and the Velocity Correction Programs are loaded as a series of independent modules. By storing a few key addresses in known absolute core locations,the location of all programs can be communicated to the various other programs. In this fasion, a very flexible system is achieved which is still easy to load.

The entire system will be operated with a two instruction waiting main program. As interrupts are generated, they will initiate SYMBOL programs, and if compute codes are inputed, control will be transferred to the FORTRAN programs. Somewhere at the end of the FORTRAN program a Branch Unconditional indirect instruction (BRU*) will be inserted to disconnect the interrupt and return control to the main program loop.

CONCLUSION

The programming technique presented here offers significant advantages in a certain clss of real time systems. In any system where the computer has an adequate amount of time to do the required computations so that the reduced computing speed of a FORTRAN coded program is not objectionable, and where the amount of mathematical and logical computation is large compared with the programming required to interface with external equipment, this technique is applicable and will save a great deal of programming time. It would be particularly useful in an experimental control application where the master program is written by someone intimately familiar with the computer, and equipment, but the computer has to be used by experimenters who are not so familiar with computers and programming. The experimenters would then only have to learn FORTRAN and write their programs in that languate instead of going to the considerable extra effort in learning and using SYMBOL. This technique would also be useful in an application where time of execution was a critical factor. The program could initially be written in FORTRAN and then, after all the logic and mathematics were worked out, the program could be coded in SYMBOL. A study of execu- times might reveal that certain portions were critical and only these would have to be rewritten.

When completed, the Ames midcourse guidance simulation will be used for a number of studies. Optical instrument and navigation techniques will be investigated in an effort to evaluate such equipment. Control and display techniques for a spacecraft computer will be studied both from a hardware and an operational viewpoint. Velocity correction techniques, attitude stabilization, automatic and manual control systems, abort techniques, and operational procedures, are some of the possible subjects for study using the simulation as a working base. It is also planned to put the control panel in a three-man, full-scale lunar mission capsule, and run simulated long-duration lunar missions in a realistic spacecraft environment.

REFERENCES

1. McLean, John D., Schmidt, Stanley F., and McGee, Leonard A.: Optimal Filtering and Linear Prediction Applied to a Midcourse Navigation System for the Circumlunar Mission. NASA TN D-1208, 1962.

2. Smith, Gerald L., Schmidt, Stanley F., McGee, Leonard A.: Application of Statistical Filter Theory to the Optimal Estimation of Position and Velocity Onboard a Circumlunar Vehicle. NASA TR R-135, 1962.

3. Smith, Gerald L.: Multivariable Linear Filter Theory Applied to Space Vehicle Guidance. Paper presented at SIAM Symposium on Multivariable System Theory, November 1962.

A Digital Lunar Midcourse Guidance Simulation

By David Kipping

CAPTIONS

Fig. 1 -- Laboratory for Guidance and Control Systems

Fig. 2 -- SDS 920 Digital Computer System

Fig. 3 -- Spacecraft Computer Control and Display Panel

Fig. 4 -- Moving Base Star/Planet Simulation

Fig. 5 -- Midcourse Guidance Simulation

Fig. 6 -- Computer System Program; Clock and Timing

Fig. 7 -- Computer System Program; Data Input and Display

Fig. 8 -- Midcourse Control and Display Program

Fig. 9 -- Velocity Correction Simulation

Fig.10 -- Schematic Diagram of Midcourse Guidance Scheme

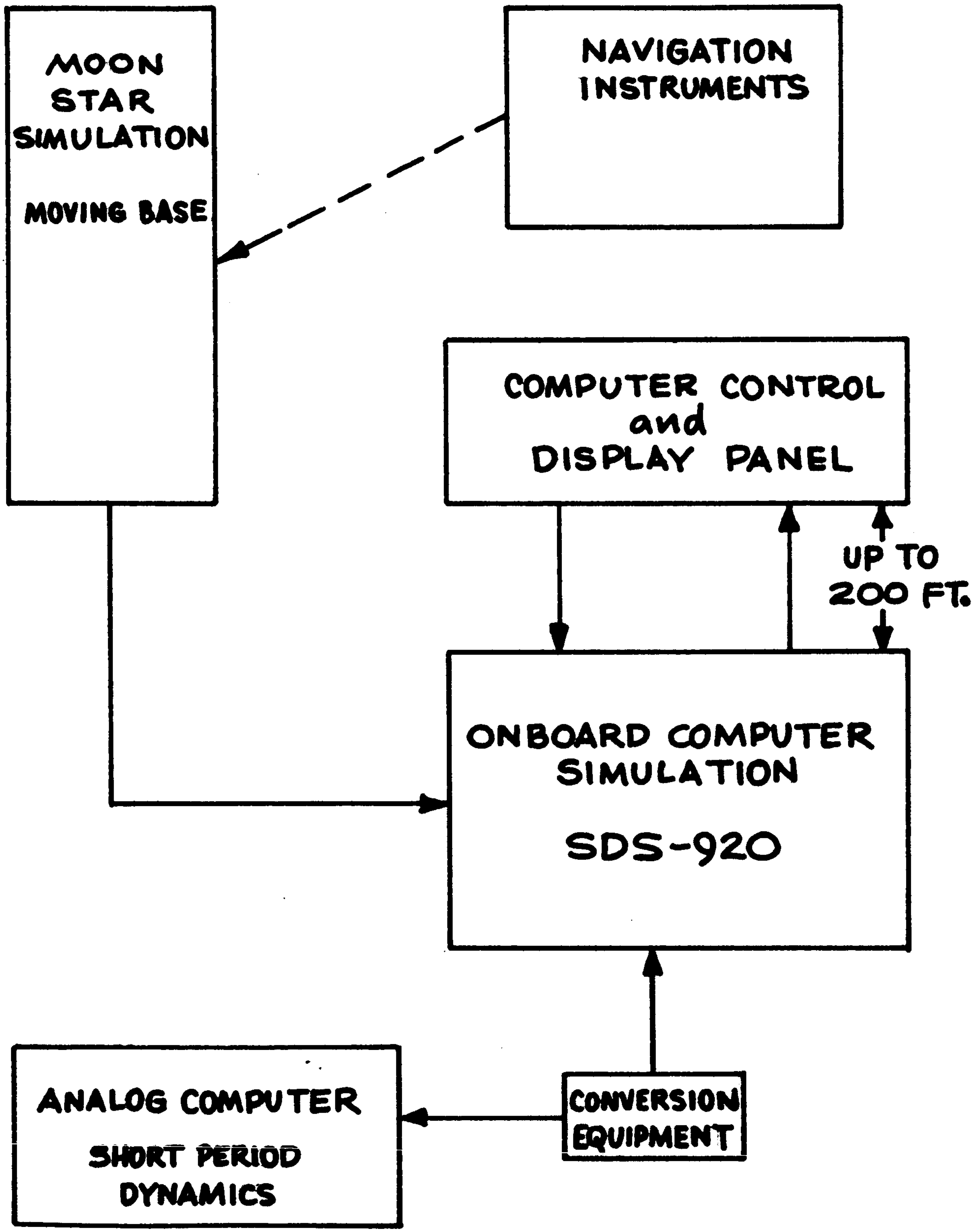
Fig.11 -- Optimal State Estimation Scheme

Fig.12 -- Midcourse Guidance Program

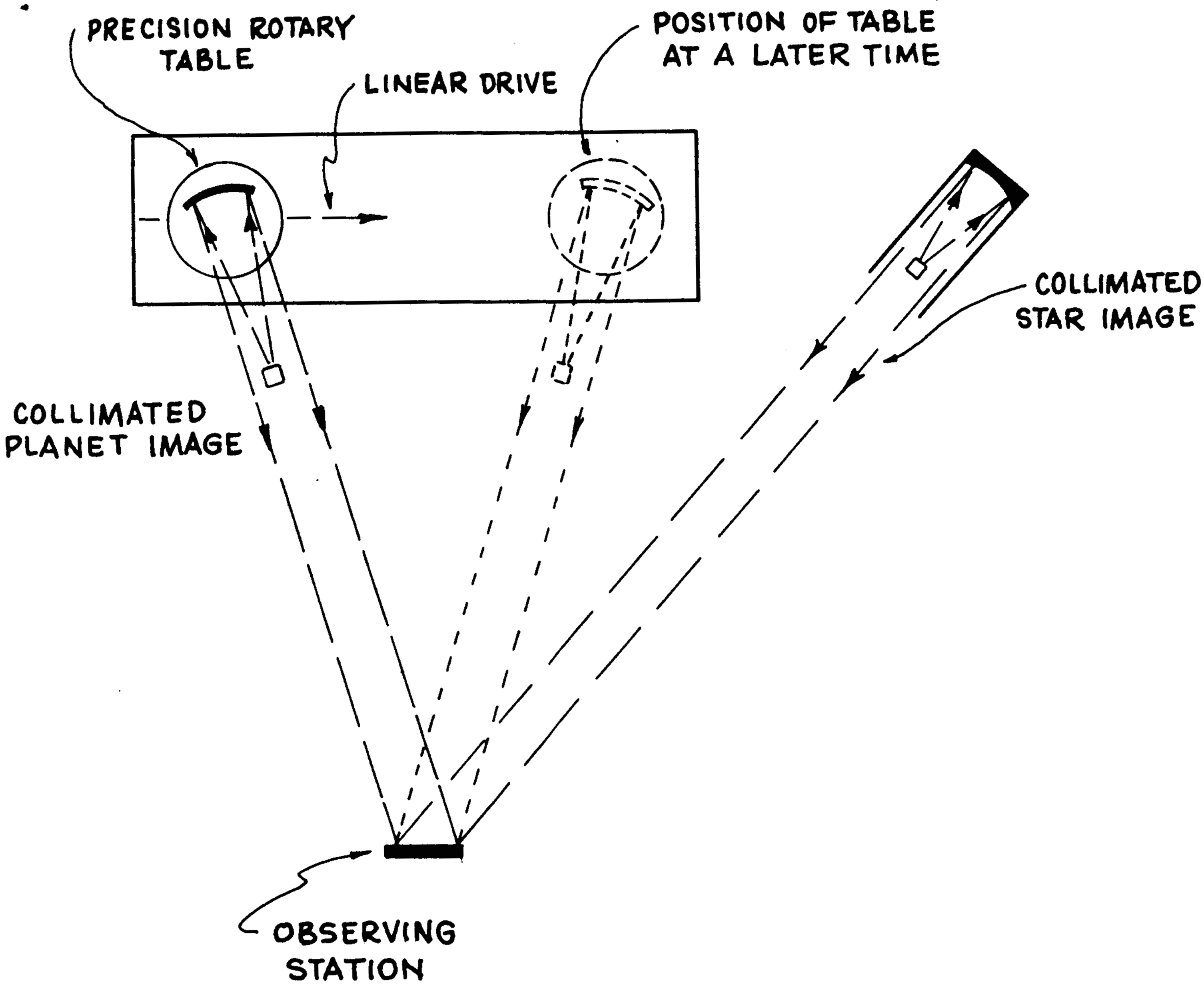
Fig.13 -- Midcourse Guidance Simulation Storage Requirements

ACKNOWLEDGMENT

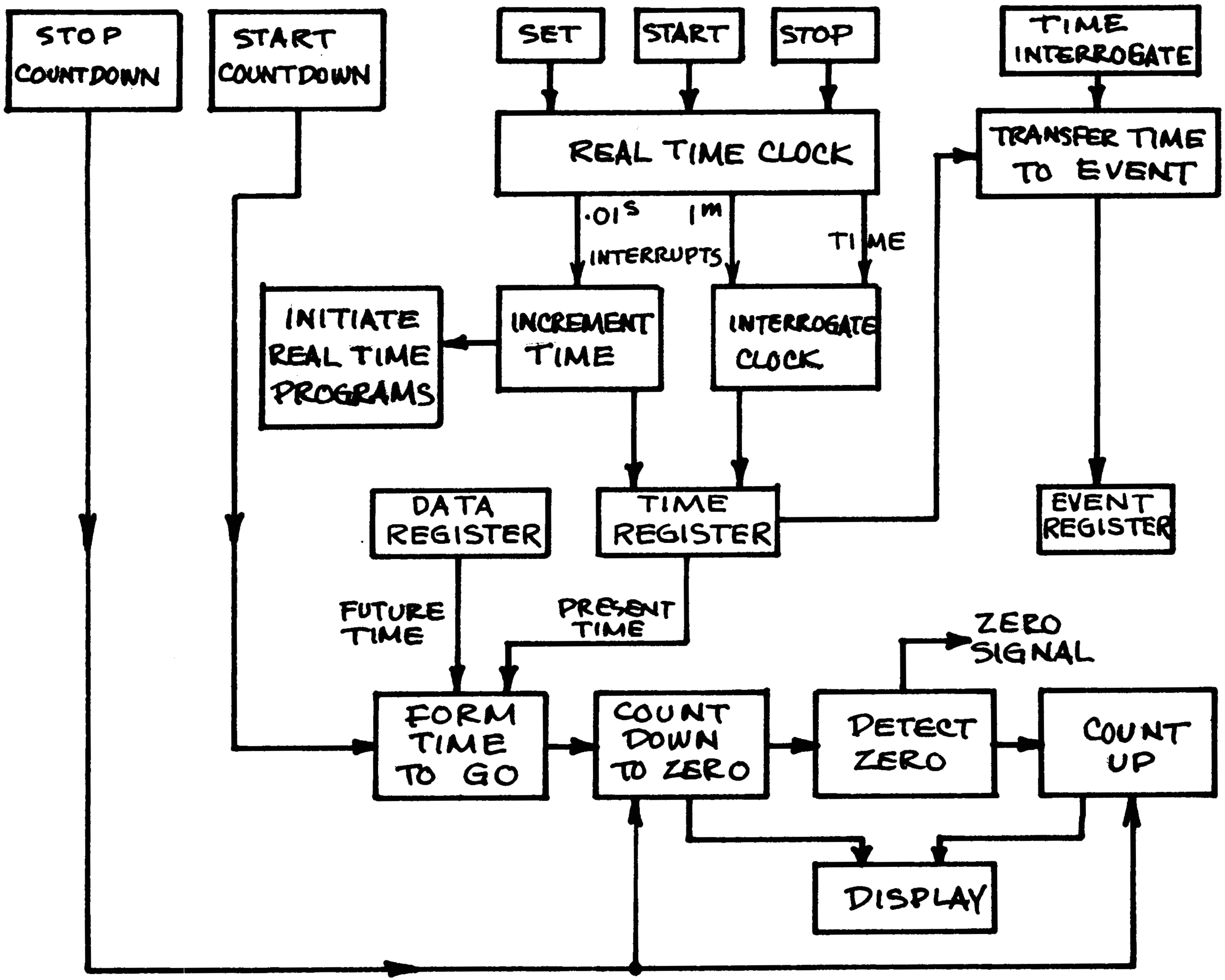
The computing system described in this paper including the Control and Display Panel was designed and built by Scientific Data Systems to detailed hardware and functional specifications written by Ames Research Center. The Computer System Program, and the program to provide compatibility with interrupts, were both written by Scientific Data Systems to a detailed functional specification. All other programming was done in-house by Ames Research Center.



LABORATORY FOR GUIDANCE
AND CONTROL SYSTEMS

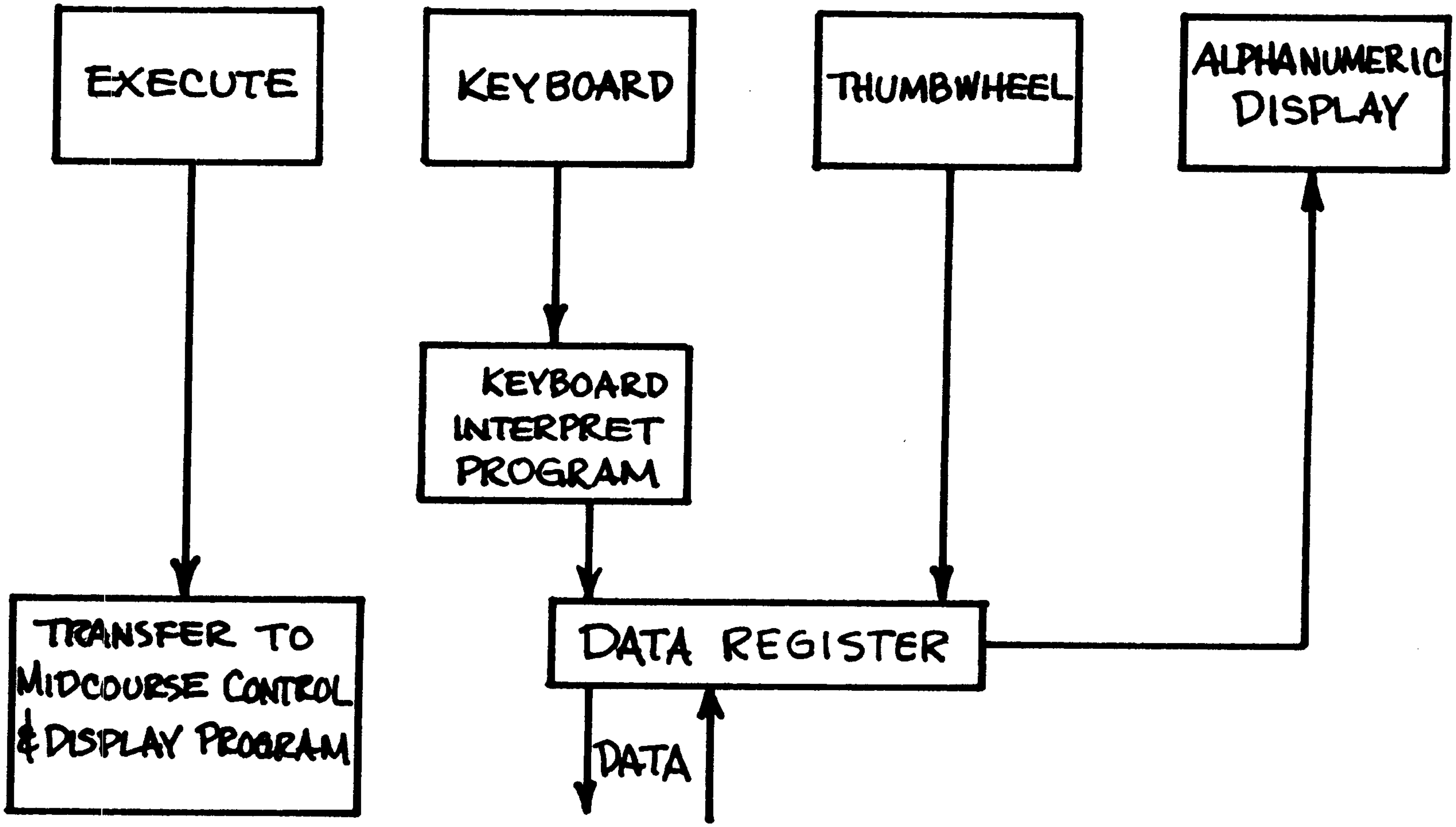


MOVING BASE STAR/PLANET SIMULATION

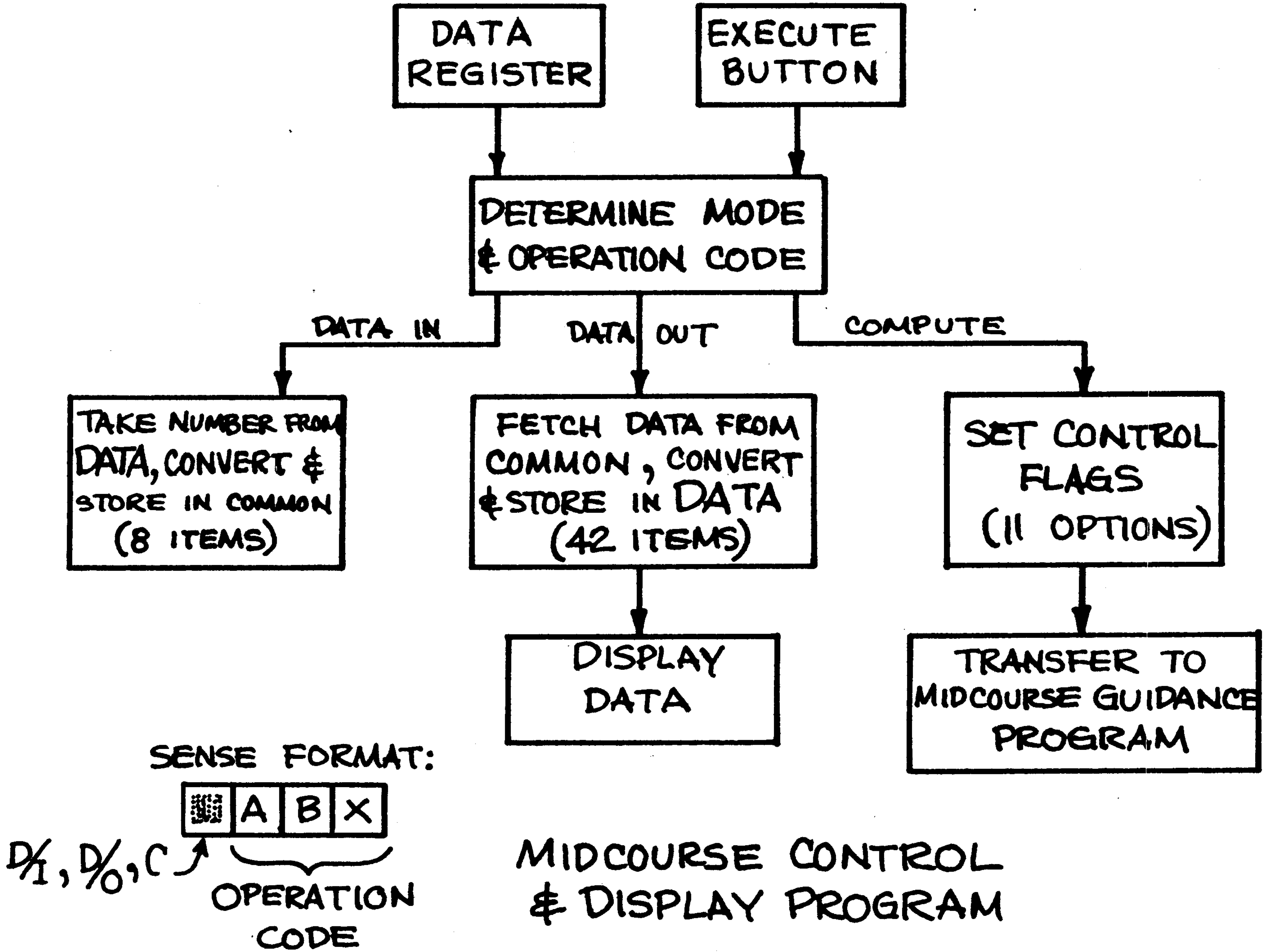


COMPUTER SYSTEM PROGRAM
CLOCK & TIMING

9

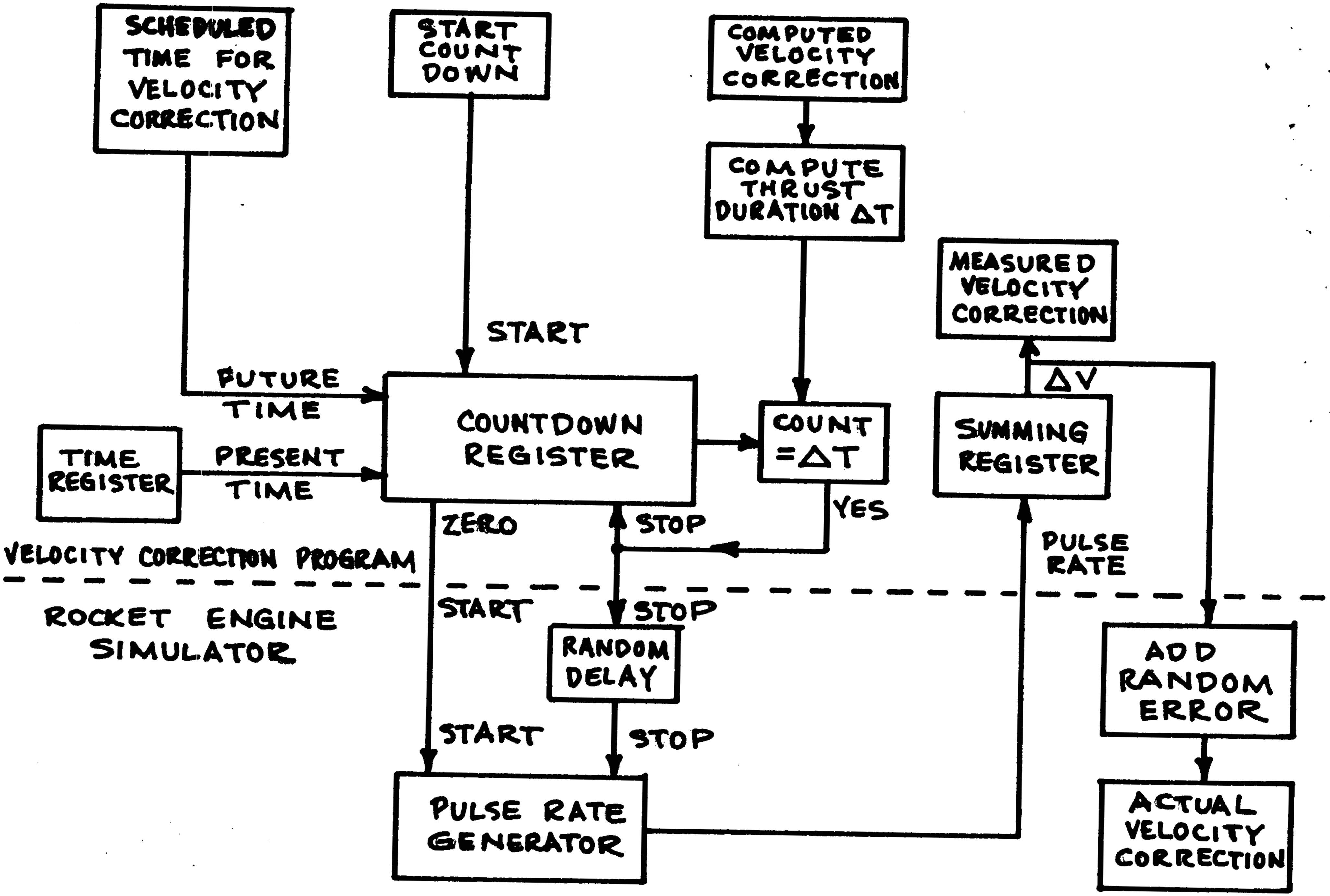


COMPUTER SYSTEM PROGRAM
DATA INPUT & DISPLAY

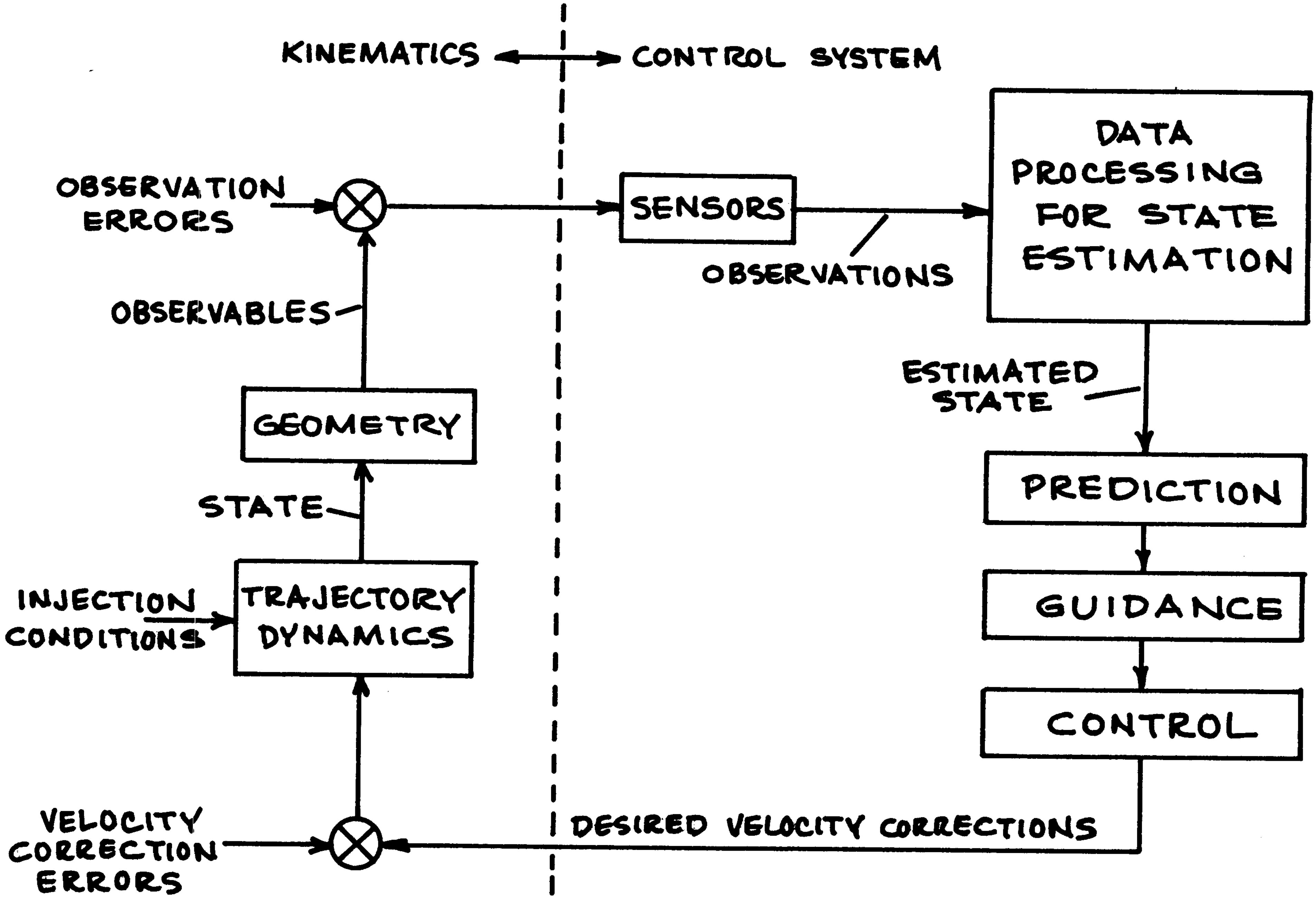


MIDCOURSE CONTROL & DISPLAY PROGRAM

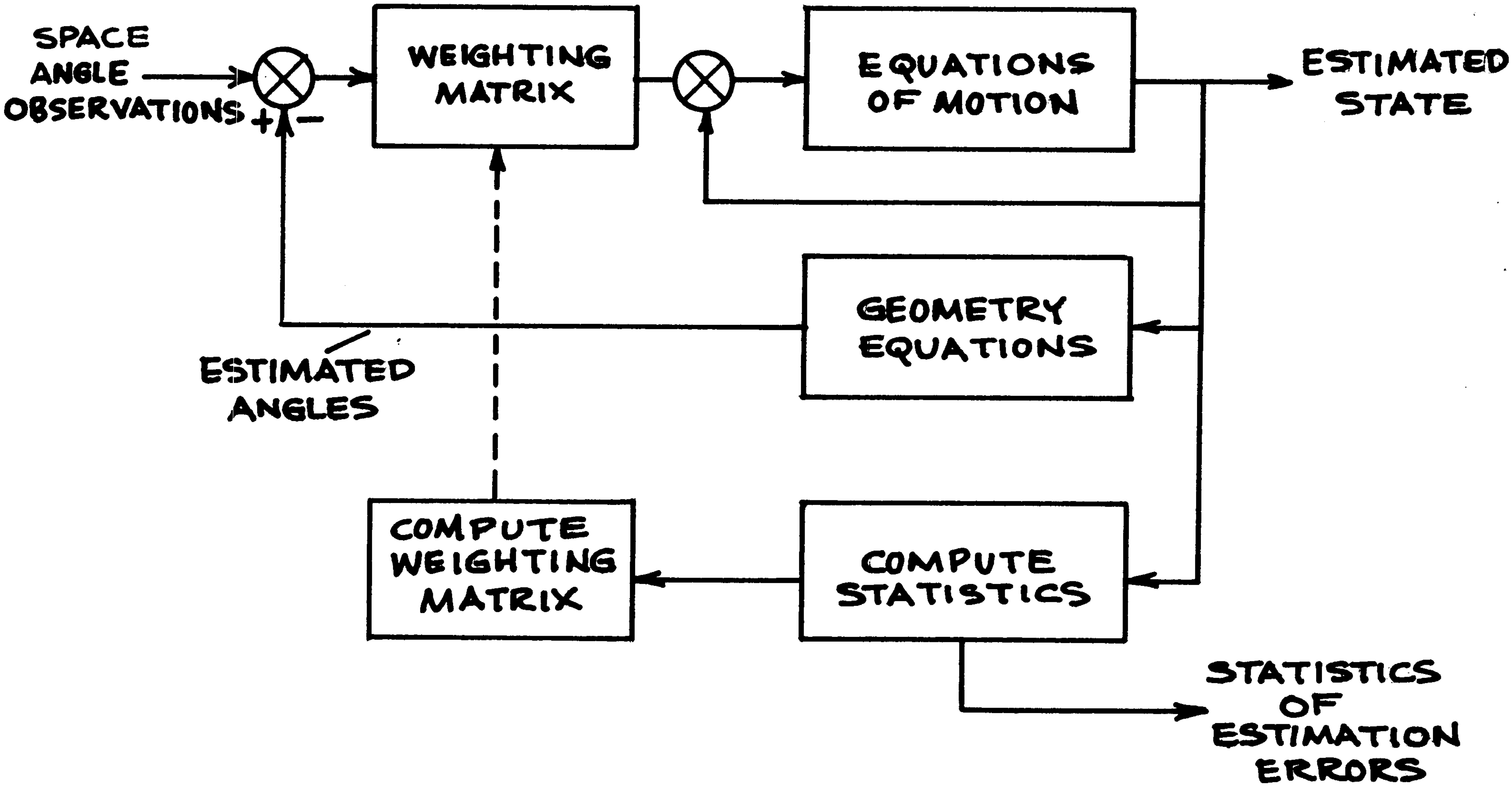
9



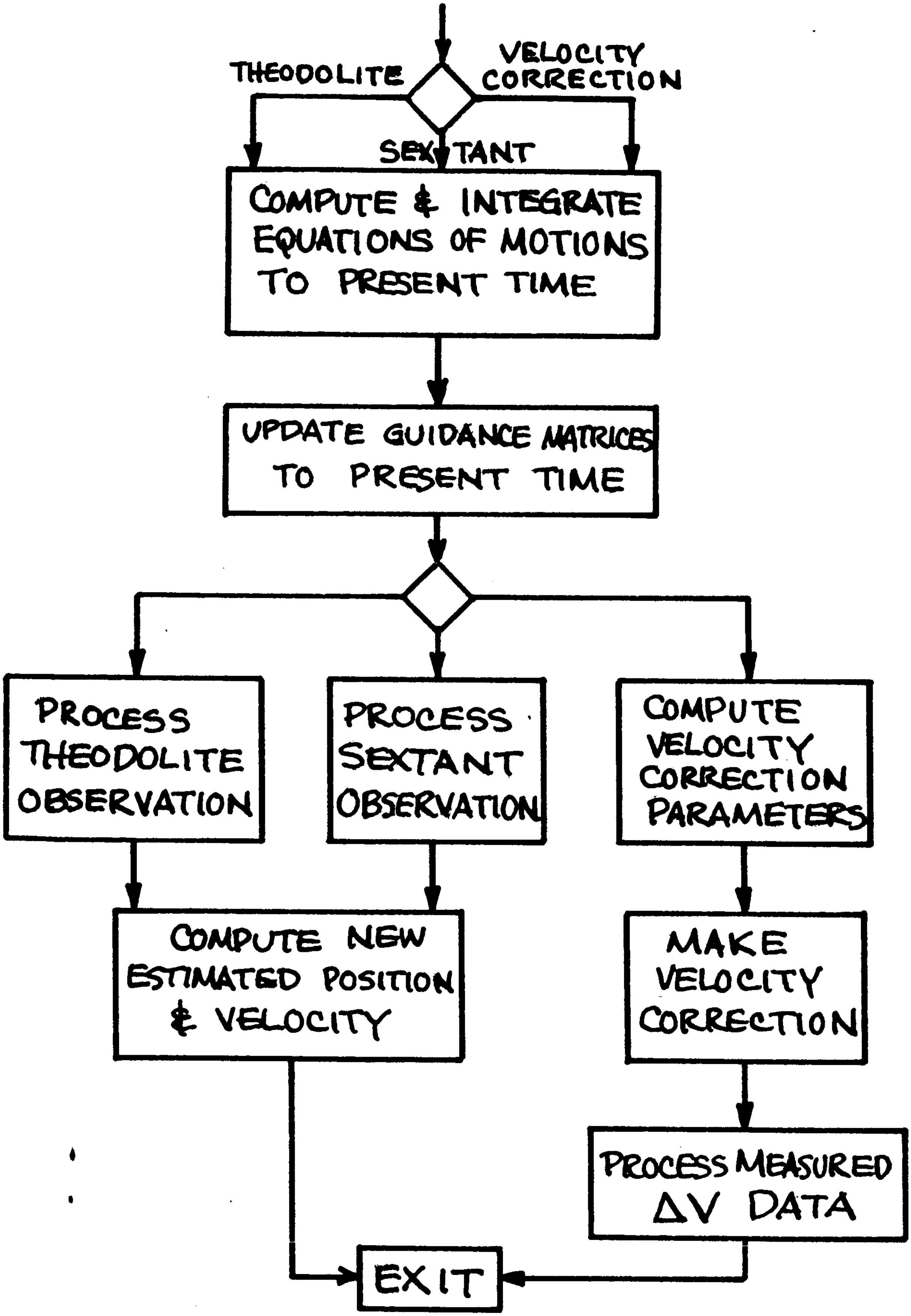
VELOCITY CORRECTION SIMULATION



SCHEMATIC DIAGRAM OF A MIDCOURSE GUIDANCE SYSTEM



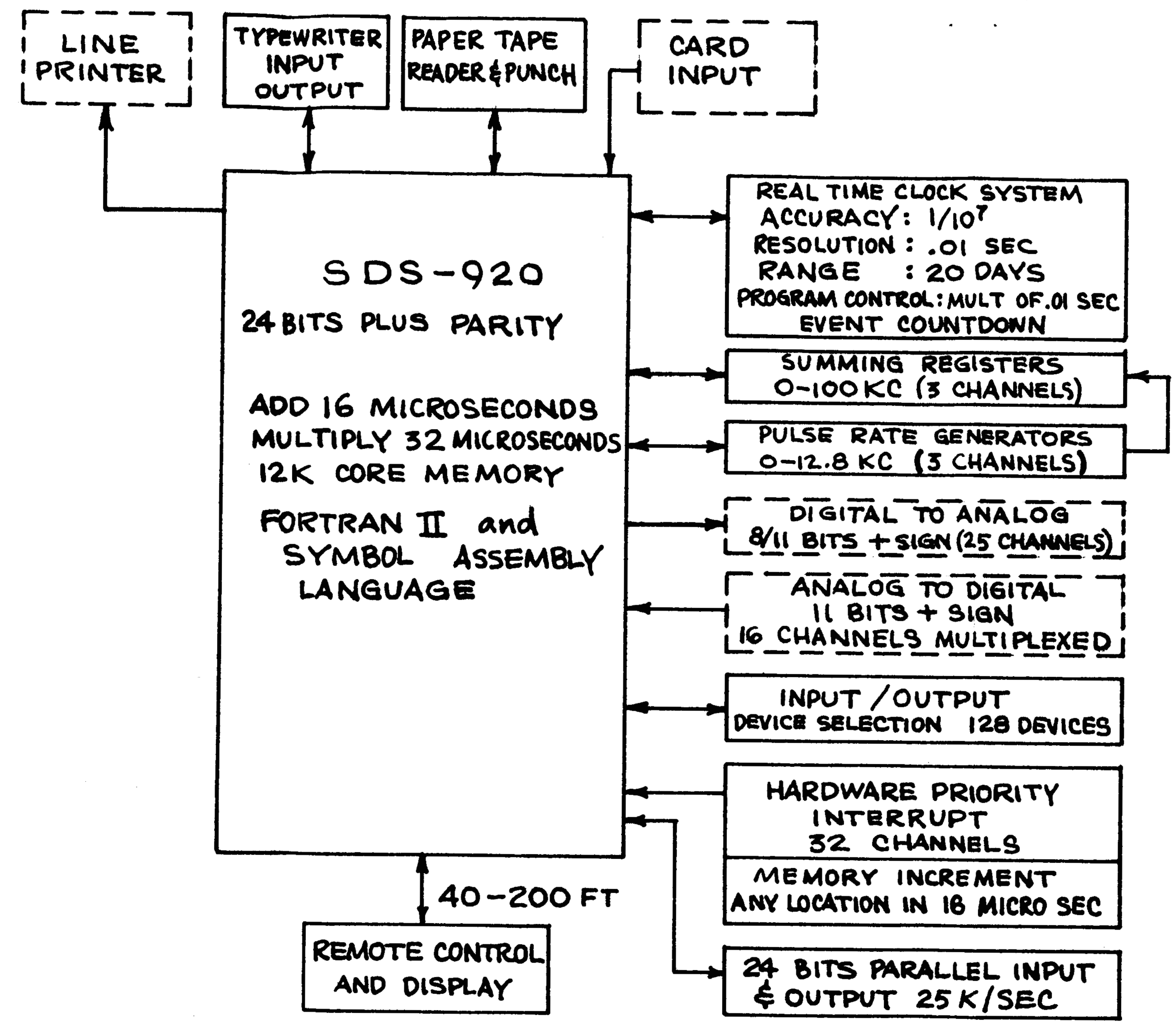
OPTIMAL STATE ESTIMATION SCHEME



MIDCOURSE GUIDANCE PROGRAM

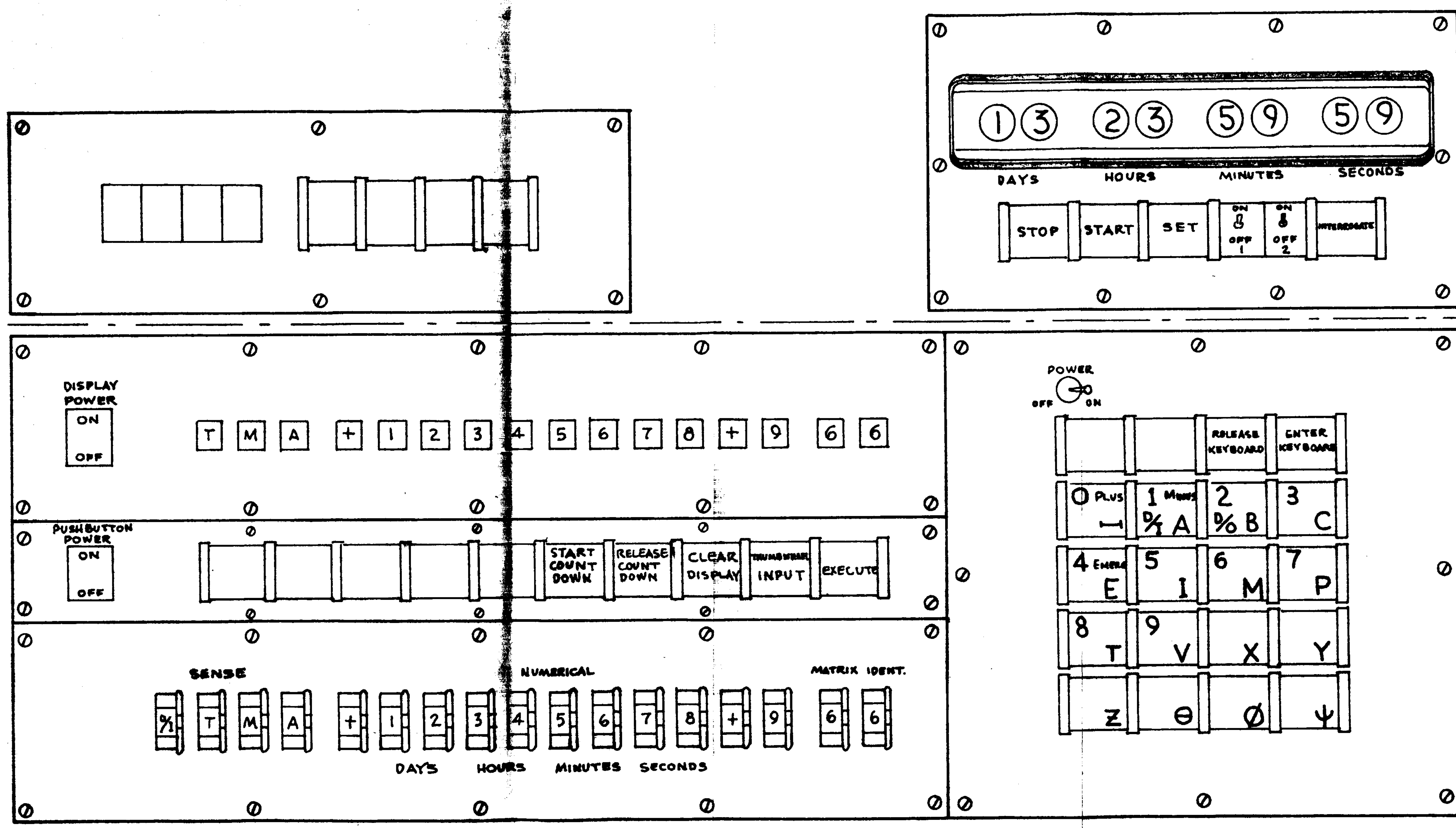
ITEM	IBM 7090		SDS 920 (est)	
	SIMULATION	ONBOARD	SIMULATION	ONBOARD
SUBROUTINES WRITTEN BY G&CS	7307	4944	5070	3483
LIBRARY SUBROUTINES	1905	1713	1713	1713
DATA STORAGE	2010	1635	1934	1900
FORTRAN				
SUBROUTINES	4354	2000	1300	1000
MONITOR	1914	0	0	0
TOTAL	17,490	10,292	10,017	8096

MIDCOURSE GUIDANCE PROGRAM
STORAGE REQUIREMENTS

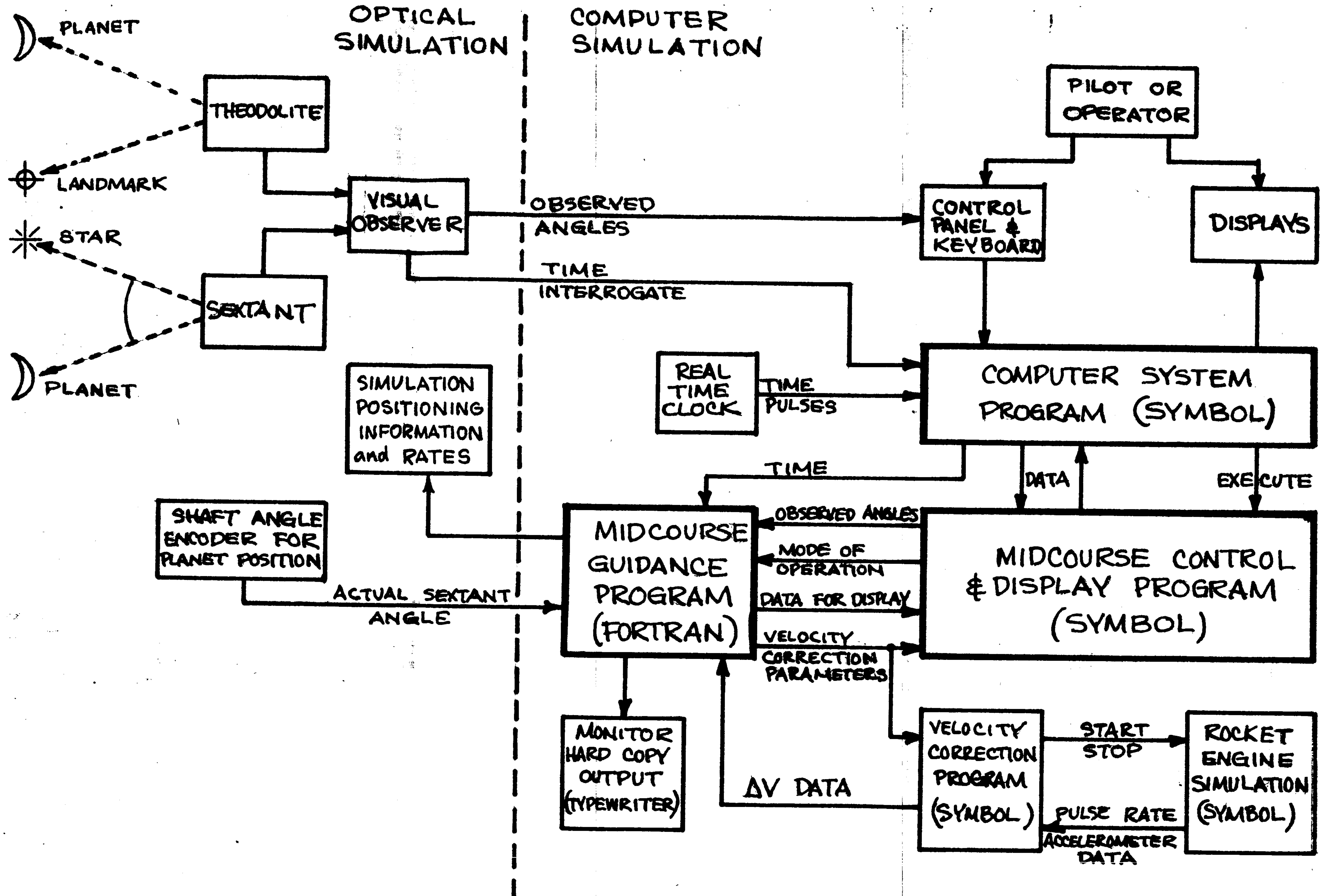


SDS 920 DIGITAL COMPUTER SYSTEM

③



SPACECRAFT COMPUTER CONTROL AND DISPLAY PANEL



MIDCOURSE GUIDANCE SIMULATION